Final Thoughts

George W. Dinolt
CS4605

Introduction

- Security Policies
- The Goal
- The Approach
- The Model Framework
- Models
- System Architecture
- Current State of Affairs

Security Policies

- Policy is a verbal description of allowed and/or disallowed information flow
- Policy may be Mandatory or Discretionary Access Control
- Policy may be for information privacy or integrity
- Policy may provide Provision of Service Guarantees

The Goal

Provide a way of precisely describing the access control properties of a system. Provide a process that will provide high assurance that the access control properties will be enforced by the system

The Approach

- Security Policy
- Security Model
- Formal Top Level Specifications
- Detailed Top Level Specifications
- Implementation
- CM, Distribution, Test Plans, CONOPS, Upgrade Plans, all controlled through Formal Specifications

The Model Framework

- Define State
- Define Secure State
- Define Transform from State to State
- Prove Transform takes a Secure State to a Secure State
- System modeled as a sequence
- As a result a system starting in a Secure State and only does Transforms will stay in a Secure State

State

- Subjects
- Objects
- Access Modes
- Labels
- Accesses → $\{a : a = (subject,\ object,\ mode)\}$
- States → $\{s : s \in 2^{Accesses}\}$ (s is a $State$ if s consists of a subset of $Accesses$
- State sometimes represented as an access matrix
- Transform adds/subtracts elements to/from a state

Models of Policies 1

- Secure Access defined based on properties of

$$(subject,\ object,\ mode)$$

- Allowed Accesses is the set of all Secure Accesses
- A state (the current access set) is secure if it is a subset of the Allowed Accesses
- Policy is captured by the Allowed Accesses
- Security at any time is completely known by looking at the current state

Models of Policies 2

- Secure Access defined based on properties of

  $$(subject,\ object,\ mode)$$

  and

- Secure Access based on history of accesses up to the point of the state (example, High Watermark policy)

- Secure State is includes the history

Models

- High Watermark
- Bell & LaPadula
- Biba Integrity
- RBAC (Integrity/Confidentiality)
- Clark Wilson
- Non Interference

System Architecture

- Kernel enforces Policy
- Reference Model - Always invoked, not changeable, correct

Other Models

- Protocol Analysis - Authentication, Secret Sharing, Integrity
- Models of Source Code
- Models of Composition

Current State

- We can handle stand-alone systems pretty well, but
- How do we handle Subject/Object creation/deletion (Secure Administration)?
- How do we handle label changes?
- How do we model policies that are functions of time (sequence of events)?
- How do we build systems from smaller components (composition of policies) ?
- How do we handle refinement/abstraction?

At the Beginning

?